

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

Method and Apparatus for Multi-Level Signaling

Inventors:

Mark A. Horowitz

Scott C. Best

William F. Stonecypher

1 **TECHNICAL FIELD**

2 The present invention relates to multi-level signaling and, more particularly,
3 to methods and circuits that encode multi-level signals to minimize the current
4 fluctuations between successive data transmissions.

5
6 **BACKGROUND**

7 In a typical binary signaling system, each conductor in the transmission
8 system can have one of two states: zero or one. In this type of system, each
9 conductor transmits one bit of data at a time. A multi-level signaling system
10 transmits data across a multiple conductors using several possible signal levels on
11 each conductor. For example, a 4-PAM (4-level Pulse Amplitude Modulation)
12 transmission system provides four different signal levels (i.e., four different
13 symbols) on each conductor instead of two levels, as used in a binary system. The
14 use of four different signal levels allows two bits of data to be transmitted across a
15 single conductor simultaneously.

16 In a 4-PAM transmission system that uses current-based output drivers, the
17 four different signal levels are represented by different current values. For
18 example, the four different current levels may be identified as $0i$, $1i$, $2i$, and $3i$.
19 Similarly, in a 4-PAM transmission system that uses voltage-based output drivers,
20 the four different signal levels are represented by different voltage values. For
21 example, the four different voltage levels may be identified as $0v$, $1v$, $2v$, and $3v$.
22 These drivers are usually connected to a transmission line environment that
23 presents an effective resistance or impedance to the output driver. This impedance
24 causes the output voltage to change if a current driver output changes, and causes
25

1 the output current to change if the value of the voltage driver changes.

2 These multi-level signals can be used in transmission systems that contain
3 either differential pairs of signals, or a single-ended signal referenced to ground.
4 In a transmission system utilizing many single-ended multi-level current-based
5 transmitters, it is desirable to maintain the total signal current required to transmit
6 a byte of data (or code word) at a relatively constant current level in comparison to
7 other bytes of data (or code words). If the signal current fluctuates greatly from
8 one byte to the next, the change in current flows through the power supply
9 connections and causes noise on the integrated circuit. This current change occurs
10 when using either voltage drivers or current drivers. The noise on the power
11 supply increases in systems that have a high data transmission rate and fast edge
12 rate transmitters. This noise on the power supply degrades the voltage margin of a
13 particular transmission.

14 The question of coding a number to keep the output current constant (DC
15 Balanced) has been studied for binary signals and various codes have been
16 developed for these binary signals. The IBM 8b-10b code is an example of this
17 type of code. The situation for multi-level signals is different than binary signals,
18 and thus requires a different approach to coding.

19 An improved architecture and method described herein addresses these and
20 other problems by encoding multi-level signals to minimize the current
21 fluctuations between successive data transmissions.
22
23
24
25

SUMMARY

By encoding bytes of data, the system and method discussed herein is able to maintain a relatively constant current requirement between successive data transmissions, thereby reducing the noise on the power supply driving the transmission system. Several different encoding techniques are discussed herein for encoding bytes of data to reduce current fluctuations from one data transmission to the next.

In a particular embodiment, data is transmitted on a multi-conductor signal path. Such data transmission produces current flow based on the value of the data transmitted. The change in current flow between successive data transmissions is reduced by encoding the data values, which are represented by sets of N bits, to produce corresponding sets of M symbols. Each set of M symbols represents multiple bits. Each set of M symbols is selected to produce a current flow within a predetermined range of current flows.

In another embodiment, multi-level signaling is performed by determining whether signal current required to transmit a data element is within a predetermined range of current values. The data element is encoded if the current required to transmit the data element is not within the predetermined range of current values. The data element is encoded such that the signal current required to transmit the data element is within the predetermined range of current values.

Another embodiment encodes a set of bits for transmission in a system that supports at least four discrete signal levels on a transmission medium. A first group of N bits of the set of bits are encoded using a first encoding scheme to generate a first encoded bit set that includes at least N+1 bits such that the first

1 encoded bit set has a first weighting corresponding to state of the at least N+1 bits
2 of the first encoded bit set. A second group of N bits of the set of bits is encoded
3 using a second encoding scheme to generate a second encoded bit set that includes
4 at least N+1 bits such that the second encoded bit set has a second weighting
5 corresponding to state of the at least N+1 bits of the second encoded bit set. When
6 a combination of the first weighting and the second weighting is within a
7 predetermined weighting range, the first and second encoded bit sets are combined
8 to produce an output symbol set. When the combination of the first weighting and
9 the second weighting is outside the predetermined weighting range, 1) a second
10 group of N bits of the set of bits is encoded using a third encoding scheme to
11 generate a third encoded bit set that identifies at least N+1 bits, such that the third
12 encoded bit set has a third weighting that is different from the second weighting,
13 and 2) the first and third encoded bit sets are combined to produce the output
14 symbol set.

15 **BRIEF DESCRIPTION OF THE DRAWINGS**

17 Fig. 1A is a block diagram illustrating an embodiment of a data
18 transmission system using current-based output drivers.

19 Fig. 1B is a block diagram illustrating an alternate embodiment of a data
20 transmission system.

21 Fig. 2 illustrates an example of an encoding circuit that encodes multi-level
22 signals in a manner that minimizes current fluctuations between adjacent bytes of
23 data.

24 Fig. 3 illustrates an exemplary decoding circuit that decodes the signals
25 generated by the encoding circuit shown in Fig. 2.

1 Fig. 4 illustrates an example LSB decoder used in the decoding circuit
2 shown in Fig. 3.

3 Fig. 5 illustrates an example MSB decoder used in the decoding circuit
4 shown in Fig. 3.

5 Fig. 6 is a flow diagram illustrating a procedure for transmitting data with
6 minimal current fluctuations between adjacent data transmissions.

7 Fig. 7 is a flow diagram illustrating another procedure for encoding bytes of
8 data to minimize current fluctuations from one byte of data to the next.

9 Fig. 8 illustrates a system capable of implementing the encoding procedure
10 discussed with respect to Fig. 7.

11 Fig. 9 illustrates a system capable of decoding data that was encoded by the
12 system shown in Fig. 8.

13 **DETAILED DESCRIPTION**

14
15 An improved architecture and method is discussed herein for encoding
16 bytes of data to reduce the current fluctuations between transmission of successive
17 bytes of data. In particular, the bytes of data are encoded in such a manner that the
18 total current required to transmit bytes of data is similar from one byte of data to
19 the next. A receiver of the data transmission has a decoding system that is capable
20 of decoding the encoded bytes of data to recreate the original data stream.

21 Although particular examples are discussed herein using a four-level
22 signaling system (including a 4-PAM system), the teachings of the present
23 invention can be applied to any multi-level signaling system (such as a six-level or
24 an eight-level transmission system).

1 Encoding the byte of data (or code word) before it is transmitted provides
2 for an effective control of the signaling current used to transmit bytes of data.
3 This increases the signal/noise ratio of the system, allowing one to make further
4 engineering tradeoffs such as improved noise margin, increased communication
5 distance, or greater number of bussed devices. The systems and methods
6 described herein also provide a low-latency encoding scheme.

7 Fig. 1A is a block diagram illustrating an embodiment of a data
8 transmission system 100 using current-based output drivers. The data
9 transmission system 100 is implemented as a 4-PAM system. The transmission
10 system 100 includes five conductors between an encoder 102 and a decoder 110.
11 The fifth conductor is used to transmit information identifying how the byte of
12 data was encoded, thereby allowing the decoder 110 to properly decode the
13 received data. The encoder 102 receives bytes of data on four two-bit input
14 conductors labeled IN1, IN2, IN3, and IN4. These eight bits are organized as four
15 MSB and four LSB, which are encoded into a set of five 4-PAM signals. The
16 encoder 102 encodes the received bytes of data to reduce fluctuations in the total
17 current required to transmit adjacent bytes of data. The output of the encoder is
18 provided to five multi-level output drivers 104, which are coupled to a channel
19 106 containing five conductors. In a particular embodiment, channel 106 is a
20 "Direct Rambus Channel" developed by Rambus Inc. of Los Altos, California.
21 The Direct Rambus Channel connects memory devices to other devices such as
22 microprocessors, digital signal processors, graphics processors and ASICs (not
23 shown). The opposite end of channel 106 is coupled to five multi-level receivers
24 108, which receive the transmitted data. The received data is provided to a

1 decoder 110, which decodes the received bytes of data. The decoder 110 generates
2 an output on four conductors labeled OUT1, OUT2, OUT3, and OUT4, where
3 each conductor represents a 2-bit value. The output of decoder 110 matches the
4 input to the encoder 102.

5 Although particular implementations discussed herein use a 4-PAM system,
6 the teachings herein may be applied to any X-PAM system where $X \geq 4$. For
7 example, in an 8-PAM system, eight conductors are used, each of which represents
8 a 3-bit value (LSB, middle bit, and MSB). In this example, a separate weighting is
9 associated with each of the three bits.

10 As mentioned above, the encoding requirements for multi-level signaling is
11 different than binary signaling, and thus requires a different approach to coding.
12 In binary signaling, all signals have the same weight (they are either 0 or 1), so
13 one possible goal of a code which minimizes transients is to generate a code which
14 has a nearly constant number of '1's. However, in multi-level coding, the
15 situation is different since some of the bits have higher current weights than
16 others. For a binary-coded 4-level PAM bit, the most significant bit (MSB) of
17 each symbol has twice the weight of the least significant bit (LSB). For these
18 multi-level signals, the goal is to keep the sum of the output current nearly
19 constant, which does not correspond to maintaining the number of '1's in the
20 binary representation constant.

21 A particular embodiment describes a coding procedure used with a four-
22 level signaling system. The coding procedure provides a mapping from input data
23 to encoded output data. The mapping is performed with minimal bit manipulation.
24 An exception system is used to provide all data values for the output set of data.

1 The combination of the mapping function and the exception system provides a
2 compact implementation suitable for high-speed bussed communications. The
3 coding procedure described herein reduces the symbol sum deviation from 12i on
4 four non-encoded 4-PAM signals to 1i on five encoded 4-PAM signals. The
5 coding procedure ensures that the weighted sum of the five encoded 4-PAM
6 signals is 7i or 8i, where 15i is the maximum sum across the set of signals.

7 To encode data, the eight bit input data is separated into four MSB bits and
8 four LSB bits. This separation provides for four input symbols, each of which can
9 range in value from zero to three. As discussed below, the data encoding is
10 performed using two sub-encoders, one acting on the MSB bits as a group and the
11 other acting on the LSB bits as a group. The MSB bits have a weight of two and
12 the LSB bits have a weight of one.

13 Table 1 below illustrates an embodiment of the encoding (e.g., mapping) of
14 the MSB bits. It will be appreciated that other coding procedures or tables may be
15 used to implement the signaling system described herein.
16
17
18
19
20
21
22
23
24
25

TABLE 1

| <u>d1,c1,b1,a1</u> | <u>E1,D1,C1,B1,A1</u> | <u>MSB Weight</u> |
|--------------------|-----------------------|-------------------|
| 0 0 0 0 | 1 0 1 0 1 | 6 |
| 0 0 0 1 | 1 0 0 0 1 | 4 |
| 0 0 1 0 | 1 0 0 1 0 | 4 |
| 0 0 1 1 | 1 0 0 1 1 | 6 |
| 0 1 0 0 | 1 0 1 0 0 | 4 |
| 0 1 0 1 | 0 0 1 0 1 | 4 |
| 0 1 1 0 | 0 0 1 1 0 | 4 |
| 0 1 1 1 | 0 0 1 1 1 | 6 |
| 1 0 0 0 | 1 1 0 0 0 | 4 |
| 1 0 0 1 | 0 1 0 0 1 | 4 |
| 1 0 1 0 | 0 1 0 1 0 | 4 |
| 1 0 1 1 | 0 1 0 1 1 | 6 |
| 1 1 0 0 | 1 1 1 0 0 | 6 |
| 1 1 0 1 | 0 1 1 0 1 | 6 |
| 1 1 1 0 | 0 1 1 1 0 | 6 |
| 1 1 1 1 | 1 1 0 1 0 | 6 |

The first column of Table 1 identifies the sixteen possible input MSB combinations, represented by the four bits: d1, c1, b1 and a1. The second column of Table 1 identifies the encoded MSB value associated with the corresponding

1 input value in the first column. The encoded MSB value is represented by five
2 bits: E1, D1, C1, B1 and A1. The third column of Table 1 identifies the MSB
3 weight, i.e., the weighted sum of each of the encoded MSB values in the second
4 column. Since each of the MSB bits has a weight of two, the MSB weight is
5 calculated by multiplying the number of '1's in the encoded MSB by two (i.e., the
6 weight of the MSB bit). Since each encoded MSB value in Table 1 has either two
7 or three '1's, the MSB weight for each encoded MSB value is either four (2×2) or
8 six (3×2). For the MSB sub-encoder, it is desirable to maintain the MSB weight
9 at four or six to minimize the current fluctuations between temporally or
10 physically successive bytes of data. As further discussed below, maintaining the
11 MSB weight at four or six for all input combinations will allow the LSB sub-
12 encoder to subsequently adjust the total signal weight at $7i$ or $8i$.

13 The mapping function shown in Table 1 results in a majority of the bits
14 passing through the mapping function unchanged. In most cases, the input bits are
15 not changed by the mapping function. The additional encoded MSB bit (E1) is
16 utilized to create two or three '1's in the encoded MSB value. The use of the E1
17 bit is sufficient to adjust all input values to within the weighted sum target of 4 or
18 6 with the exception of two entries (0000 and 1111). Here, two spare codes are
19 assigned which have a minimal impact to the encoding or decoding functions.
20 This is discussed further below.

21 When assigning the encoded MSB values in Table 1 above, several valid
22 combinations were not used. Specifically, {1 0 1 1 0}, {0 0 0 1 1}, {0 1 1 0 0},
23 and {1 1 0 0 1} are not used as encoded MSB values, although each unused
24 combination includes two or three '1's and, therefore, meet the requirement of a
25

weighted sum of four or six. Any of these four unused combinations could be used to handle the two exception cases or for purposes other than transmitting data, such as control characters, without violating the goals of the mapping function.

Table 2 below illustrates an example of the encoding (e.g., mapping) of the LSB bits.

TABLE 2

| <u>d0,c0,b0,a0</u> | <u>E0,D0,C0,B0,A0</u> | <u>LSB Weight</u> | <u>Inverted E0,D0,C0,B0,A0</u> | <u>Inverted LSB Weight</u> |
|--------------------|-----------------------|-----------------------|------------------------------------|------------------------------------|
| 0 0 0 0 | Exception | | | |
| 0 0 0 1 | 0 0 0 0 1 | 1 | 1 1 1 1 0 | 4 |
| 0 0 1 0 | 0 0 0 1 0 | 1 | 1 1 1 0 1 | 4 |
| 0 0 1 1 | 0 0 0 1 1 | 2 | 1 1 1 0 0 | 3 |
| 0 1 0 0 | 0 0 1 0 0 | 1 | 1 1 0 1 1 | 4 |
| 0 1 0 1 | 0 0 1 0 1 | 2 | 1 1 0 1 0 | 3 |
| 0 1 1 0 | 0 0 1 1 0 | 2 | 1 1 0 0 1 | 3 |
| 0 1 1 1 | 1 1 0 0 0 | 2 | 0 0 1 1 1 | 3 |
| 1 0 0 0 | 0 1 0 0 0 | 1 | 1 0 1 1 1 | 4 |
| 1 0 0 1 | 0 1 0 0 1 | 2 | 1 0 1 1 0 | 3 |
| 1 0 1 0 | 0 1 0 1 0 | 2 | 1 0 1 0 1 | 3 |
| 1 0 1 1 | 1 0 1 0 0 | 2 | 0 1 0 1 1 | 3 |
| 1 1 0 0 | 0 1 1 0 0 | 2 | 1 0 0 1 1 | 3 |

| | | | | | |
|---|---------|-----------|---|-----------|---|
| 1 | 1 1 0 1 | 1 0 0 1 0 | 2 | 0 1 1 0 1 | 3 |
| 2 | 1 1 1 0 | 1 0 0 0 1 | 2 | 0 1 1 1 0 | 3 |
| 3 | 1 1 1 1 | 1 0 0 0 0 | 1 | 0 1 1 1 1 | 4 |

The first column of Table 2 identifies the sixteen possible LSB combinations, represented by the four bits: d0, c0, b0 and a0. The second column of Table 2 identifies the encoded LSB value associated with the corresponding four-bit LSB combination. The encoded LSB value is represented by five bits: E0, D0, C0, B0 and A0. The encoded LSB value associated with the four-bit LSB combination “0000” is handled in a special manner using the exception system discussed below. The third column of Table 2 identifies the LSB weight, which is the sum of the encoded LSB values in the second column (i.e., the sum of the ‘1’s in the encoded LSB value). The fourth column of Table 2 identifies the inverted LSB value (i.e., the inverse of the bits shown in the second column of Table 2). The fifth column identifies the weight of the inverted LSB value. As shown in Table 2, the LSB weight (column three) is one or two and the inverted LSB weight is three or four.

As discussed earlier, it is desirable to maintain a weighted sum for an encoded word of either 7i or 8i, where the word sum is the sum of the MSB weight and the LSB weight. As shown in Table 1, the MSB encoding function outputs {E1, D1, C1, B1, A1} produce a weighted sum of four or six. Thus, to maintain a word sum of 7i or 8i, the encoded LSB bits {E0, D0, C0, B0, A0} need to add three or four when the MSB weighted sum is four, and need to add one or two when the MSB weighted sum is six. As shown in Table 2, all of the LSB weights

1 (except 0000) have a resulting weighted sum of one or two so will directly add to
2 the MSB weighted sum of six. When the MSB weighted sum is four, the inverse
3 of the LSB outputs is used, which results in a LSB weight of three or four.

4 The encoding function shown in Table 2 results in a majority of the bits
5 passing through the encoding function unchanged. In most cases, the input bits
6 are not changed by the encoding function. For the inputs {d0, c0, b0, a0} that
7 have more than two '1's (e.g., 0111, 1011, 1101, 1110 and 1111), the E0 bit is used
8 as an indicator that the {D0, C0, B0, A0} outputs are inverted from their
9 corresponding input values. Thus, if E0 is one, the input bits are inverted. If E0 is
10 zero, then the input bits remain unchanged. Having the E0 bit set to one in
11 combination with the remaining inverted bits provides an LSB weight of one or
12 two.

13 As mentioned above, an exception system is used to complete the LSB
14 table (Table 2) where the input value {d0, c0, b0, a0} is {0 0 0 0}. This value
15 leaves 16 of the 256 input values without a corresponding output value (i.e., the 16
16 eight bit values that end in "0000"). The exception system uses some of the
17 unused MSB output codes as "exception indicators." These exception indicators
18 are used by a decoding system to properly handle the exception cases.

19 Fig. 1B is a block diagram illustrating an alternate embodiment of a data
20 transmission system 150. Data transmission system 150 is similar to system 100
21 shown in Fig. 1A, but transmits data serially rather than in parallel. An encoder
22 152 receives bytes of data on four two-bit input conductors labeled IN1, IN2, IN3,
23 and IN4. These eight bits are organized as four MSB and four LSB, which are
24 encoded using methods described herein to reduce current fluctuations into a set of
25 five 2-bit conductors. The five conductors serve as inputs to a serializer 156 or

1 shift register. A 2-bit conductor connects the output of the serializer 156 to a
2 multi-PAM output transmitter 154 driving a communications medium 160.

3 The serial data stream is received by a multi-PAM receiver 162, converting
4 the multi-PAM input into a 2-bit output value. The multi-PAM receiver 162 is
5 connected to a deserializer 158 or shift register which accumulates five 2-bit
6 symbols (values). The output of the deserializer 158 is connected to a decoder 164
7 which is described herein. The decoder 164 generates an output on four
8 conductors labeled OUT1, OUT2, OUT3, and OUT4, where each conductor
9 represents a 2-bit value. The output of decoder 164 matches the input to the
10 encoder 152.

11 The encode and decode function taught within this invention have several
12 beneficial characteristics for serial multi-PAM communications. The data stream
13 will be DC balanced which is necessary for AC-coupled applications. Every five
14 symbols will have both a 1->0 and 0->1 transition on the MSB providing high
15 transition density for clock recovery. As well, there are several spare codes which
16 can be used for unique control characters to support a variety of communications
17 protocols.

18 Fig. 2 illustrates an example of an encoding circuit 200 that encodes multi-
19 level signals in a manner that minimizes current fluctuations between adjacent
20 bytes of data. The encoding circuit 200 receives an MSB (represented by A1, B1,
21 C1 and D1) on a four bit bus 202, and a LSB (represented by A0, B0, C0 and D0)
22 on another four bit bus 204. Bus 202 is coupled to a comparator 206, an MSB
23 encoder 208, another comparator 210, and a multiplexer 212. Comparator 206
24 determines whether the MSB is zero (i.e., all four bits are zero) by comparing the
25 MSB to a zero value. The output of comparator 206 is '0' if the MSB is non-zero

1 and the output of comparator 206 is '1' if the MSB is zero. MSB encoder 208
2 encodes the MSB into a five bit signal corresponding to the function shown in
3 Table 1 above. Comparator 210 determines whether the weighted sum of the
4 MSB bits is six. The output of comparator 210 is '0' if the weighted sum of the
5 MSB is not six, and the output of comparator 210 is '1' if the weighted sum of the
6 MSB is six.

7 Bus 204 is coupled to multiplexer 212 and a comparator 214. Comparator
8 214 determines whether the LSB is zero (i.e., all four bits are zero) by comparing
9 the LSB to a zero value. The output of comparator 214 is '0' if the LSB is non-
10 zero, and the output of comparator 214 is '1' if the LSB is zero. The output of
11 comparator 214 is coupled to the select control of multiplexer 212. Multiplexer
12 212 selects the MSB or the LSB based on the output of comparator 214. If the
13 output of comparator 214 is '1', the output of multiplexer 212 is the MSB. If the
14 output of comparator 214 is '0', the output of multiplexer 212 is the LSB.

15 A multiplexer 216 receives two different five bit signals, labeled "E1" and
16 E2". In one implementation, E1 is "10110" and E2 is "00011". The select control
17 input for multiplexer 216 is coupled to the output of comparator 206. If the output
18 of comparator 206 is '0' (MSB is non-zero), E1 is selected by multiplexer 216. If
19 the output of comparator 206 is '1' (MSB is zero), E2 is selected by multiplexer
20 216.

21 A multiplexer 218 receives five bit signals from the output of multiplexer
22 216 and the output of MSB encoder 208. The select control input for multiplexer
23 218 is coupled to the output of comparator 214, which identifies whether the LSB
24 is zero. If the output of comparator 214 is '0' (LSB is non-zero), the output of
25 MSB encoder 208 is selected by multiplexer 218. If the output of comparator 214

1 is '1' (LSB is zero), the output of multiplexer 216 is selected by multiplexer 218.
2 The output of multiplexer 218 is the encoded MSB signal.

3 The output of multiplexer 212 is coupled to the input of a LSB encoder
4 222, which encodes the LSB into a five bit signal corresponding to the function
5 shown in Table 2 above. The output of LSB encoder 222 is coupled to an inverted
6 input and a non-inverted input of a multiplexer 226. The select control input of
7 multiplexer 226 is coupled to the output of an OR gate 224, which receives inputs
8 from comparator 210 and comparator 214. The output of OR gate 224 is '1' when
9 the output of comparator 210 is '1' (weighted sum of MSB is six) or the output of
10 comparator 214 is '1' (LSB is zero) Otherwise, the output of OR gate 224 is '0'.
11 If the select control input receives '1', multiplexer 226 selects the five bit signal
12 received from LSB encoder 222. If the select control input receives '0',
13 multiplexer 226 selects the inverse of the five bit signal received from LSB
14 encoder 222 (i.e., each of the five bits is inverted).

15 A multiplexer 228 is coupled to receive the output of multiplexer 226 and a
16 signal identified as "E2L". In one implementation, E2L is "00111". The select
17 control input of multiplexer 228 is coupled to the output of an AND gate 220.
18 AND gate 220 receives inputs from the output of comparator 206 and from the
19 output of comparator 214. If both the output of comparator 206 is '1' (MSB is
20 zero) and the output of comparator 214 is '1' (LSB is zero), the output of AND
21 gate 220 is '1'. Otherwise, the output of AND gate 220 is '0'. If the output of
22 AND gate 220 is '0', multiplexer 228 selects the output of multiplexer 226. If the
23 output of AND gate 220 is '1', multiplexer 228 selects E2L. The output of
24 multiplexer 226 is the encoded LSB signal.

1 In operation, the encoding circuit 200 shown in Fig. 2 encodes the MSB
2 and LSB using the functions described in Table 1 and Table 2 above. The circuit
3 shown in Fig. 2 also handles the exception system when the LSB is "0000".
4 Specifically, when the LSB is zero, the circuit multiplexes the MSB input data into
5 the LSB input encoder 222 using multiplexer 212. Thus, when the LSB is zero,
6 the MSB bits are used in place of the LSB bits. One exception code (E1) is used
7 when the MSB is non-zero and another exception code (E2) is used when the MSB
8 is zero. These exception codes (E1 and E2) are selected by multiplexer 216.

9 As mentioned above, when the LSB is zero, the MSB bits are provided to
10 LSB encoder 222 through multiplexer 212. One of the unused MSB codes
11 (exception code E1) is used to identify the situation where the LSB is zero. A
12 particular implementation assigns the value "10110" to E1. In alternate
13 implementations, any other unused code may be substituted for "10110", with a
14 corresponding change in the decoder circuit and compensation for any change in
15 the weighted sum for E1.

16 When both the MSB is zero and the LSB is zero, swapping the MSB into
17 the LSB is not sufficient. In this situation, an additional unused MSB code (E2) is
18 driven onto the MSB outputs. When this code is received by a decoder circuit, the
19 decoder knows to drive all '0's for all outputs, regardless of the LSB value. In one
20 implementation, the value of E2 is "00011". Additionally, since the LSB values
21 are not correct, another code (E2L) is driven onto the LSB outputs each time that
22 E2 is driven onto the MSB outputs. The value of E2L is chosen such that the sum
23 of all bits transmitted is $7i$ or $8i$. In this example, the value of E2L is "00111".
24 This implementation generates a sum of $7i$ (E2 has a weight of four which is
25 added to E2L which has a weight of three).

1 When it is desirable to send control codes using the some of the spare codes
2 available, an additional input will be necessary to specify that a control character
3 is desired. The existing circuit could be easily extended such that the control
4 signal selects a new control code onto the outputs a1,b1,c1,d1,e1. The control
5 code would be selected from one of the available spare MSB values.

6 Fig. 3 illustrates an exemplary decoding circuit 300 that decodes the signals
7 generated by the encoding circuit shown in Fig. 2. The encoded MSB is received
8 on a bus 302, which is coupled to an MSB decoder 306, a first exception code (E1)
9 comparator 308, and a second exception code (E2) comparator 310. MSB decoder
10 306 decodes the received five bit signal (the encoded MSB) into a four bit MSB
11 using the function described above with respect to Table 1. First exception code
12 comparator 308 determines whether the first exception code E1 is present in the
13 encoded MSB. If the encoded MSB contains E1, the output of comparator 308 is
14 '1'. Otherwise, the output of comparator 308 is '0'. Second exception code
15 comparator 310 determines whether the second exception code E2 is present in the
16 encoded MSB. If the encoded MSB contains E2, the output of comparator 310 is
17 '1'. Otherwise, the output of comparator 310 is '0'.

18 The encoded LSB is received on a bus 304, which is coupled to an LSB
19 decoder 312. LSB decoder 312 decodes the received five bit signal (the encoded
20 LSB) into a four bit LSB using the function described above with respect to Table
21 2. The output of LSB decoder 312 is provided to an input of a multiplexer 318.
22 The other input of multiplexer 318 receives a zero value (i.e., "0000"). The select
23 control input of multiplexer 318 is coupled to the output of an OR gate 320. OR
24 gate 320 receives input signals from first exception code comparator 308 and
25 second exception code comparator 310. If the output of either comparator 308 or

310 is '1', the output of OR gate 320 is '1'. Otherwise, the output of OR gate 320 is '0'. If the output of OR gate 320 is '1' (i.e., one of the exception codes was detected in the encoded MSB), multiplexer 318 selects "0000" as the multiplexer output. If the output of OR gate 320 is '0' (i.e., no exception codes detected in the encoded MSB), multiplexer 318 selects the output of the LSB decoder as the multiplexer output. The output of multiplexer 318 is the decoded LSB.

A multiplexer 314 receives inputs from MSB decoder 306 and LSB decoder 312. The select control input for multiplexer 314 is coupled to the output of the first exception code comparator 308. If the output of comparator 308 is '1' (i.e., the E1 exception code was detected in the encoded MSB), multiplexer 314 selects the output from LSB decoder 312. Otherwise, multiplexer 314 selects the output from MSB decoder 306. The output of multiplexer 314 is coupled to an input of another multiplexer 316. The other input of multiplexer 316 receives a constant value "0000". The select control input for multiplexer 316 is coupled to the output of the second exception code comparator 310. If the output of comparator 310 is '1' (i.e., the E2 exception code was detected in the encoded MSB), multiplexer 316 selects the constant value "0000". Otherwise, multiplexer 316 selects the output from multiplexer 314. The output of multiplexer 316 is the decoded MSB.

In operation, the decoding circuit 300 shown in Fig. 3 decodes the encoded MSB and the encoded LSB using the functions described in Table 1 and Table 2 above. The circuit shown in Fig. 3 also handles the exception system discussed above. The decoding circuit 300 receives ten bits (five MSB bits and five LSB bits) from, for example, a high-speed receiver and serializer (not shown). Eight

1 bits (four MSB bits and four LSB bits) are generated as an output from decoding
2 circuit 300.

3 Since the exception codes E1 and E2 are provided on the MSB input
4 signals {e1,d1,c1,b1,a1}, these bits are compared to a possible match of an
5 exception code. A detected E1 code indicates that the decoded LSB should have a
6 value of "0000". When the E1 code is detected, the output of LSB decoder 312 is
7 multiplexed (via multiplexer 314) onto the MSB output bits, and the LSB output
8 bits are driven to "0000". When the E2 code is detected in the MSB input signals,
9 both the MSB outputs and the LSB outputs are driven to "0000". When neither
10 exception code is detected in the MSB input signals, the outputs of both the MSB
11 decoder 306 and the LSB decoder 312 are selected as the outputs of decoding
12 circuit 300.

13 In an alternate embodiment, one or more control characters may be
14 communicated using one or more of the unused codes as control character
15 identifiers. In this embodiment, a control character identifier is coupled to bus
16 302. If a control character is identified, the control character identifier generates
17 an output that selects the appropriate multiplexers in the decoding circuit 300. The
18 existence of a control code is identified on bus 302 (i.e., the MSB) and the actual
19 control code value is made available on bus 304 (i.e., the LSB). In this alternate
20 embodiment, an additional output signal is provided from the decoder in order to
21 communicate the fact that a control character was received.

22 Fig. 4 illustrates an example LSB decoder, such as LSB decoder 312 used
23 in the decoding circuit shown in Fig. 3. A multiplexer 402 has an inverted input
24 and a non-inverted input, each of which receives four bits (i.e., d0, c0, b0 and a0)
25 of the encoded LSB signal. The fifth bit (i.e., e0) of the encoded LSB signal is

1 coupled to the select control input of multiplexer 402. As discussed above, when
2 encoding the LSB, if bit e0 is '1', the remaining bits in the LSB are inverted.
3 Thus, to properly decode the encoded LSB signal, the bits of the encoded LSB
4 signal are inverted when bit e0 is '1'. Thus, if e0 is '1', multiplexer 402 selects
5 the inverted input, which causes the multiplexer to output inverted bits d0, c0, b0
6 and a0. If e0 is '0', multiplexer 402 selects the non-inverted input, which causes
7 the multiplexer to output the bits d0, c0, b0 and a0 as received (i.e., not inverted).
8 The logic function performed by multiplexer 402 is equivalent to an exclusive-OR
9 (XOR) operation.

10 Fig. 5 illustrates an example MSB decoder, such as MSB decoder 306 used
11 in the decoding circuit shown in Fig. 3. The encoded MSB bits are received on a
12 bus 502, which is coupled to a first comparator 504 and a second comparator 506.
13 The comparators 504 and 506 determine whether particular combinations of bits
14 are present on bus 502 which require special processing. A multiplexer 510
15 includes an inverted input and a non-inverted input, each of which receives two
16 bits (i.e., c1 and a1) of the encoded MSB signal. Multiplexer 510 has a select
17 control input which is coupled to the output of an OR gate 508. OR gate 508
18 receives inputs from comparators 504 and 506. The output of OR gate 508 is '1'
19 if one of the two particular combinations are detected in the encoded MSB bits.
20 Otherwise, the output of OR gate 508 is '0'. If the output of OR gate 508 is '1',
21 multiplexer 510 selects the inverted input, which causes the multiplexer to output
22 the inverted values of bits c1 and a1. Otherwise, multiplexer 510 outputs the non-
23 inverted values of bits c1 and a1. As shown in Table 1, bits b1 and d1 do not
24 change during the encoding process (i.e., in all sixteen combinations, the value of

b1 and d1 are the same after the encoding function is applied). Thus, bits b1 and d1 are not changed during the decoding process.

The two particular combinations mentioned above occur when the bits received on bus 502 are "11010" or "10101". As shown in Table 1, these two cases represent the two combinations in which the values of a1 and c1 are inverted during the encoding process. For all other combinations, the values of bits a1 and c1 are not changed as a result of the encoding process. The logic function performed by multiplexer 510 is equivalent to an exclusive-OR (XOR) operation.

Fig. 6 is a flow diagram illustrating a procedure 600 for transmitting data with minimal current fluctuations between successive data transmissions. Data is received for transmission across a channel or other transmission path (block 602). The received data is encoded using a multi-level signaling technique that reduces the current fluctuations between successive data transmissions (block 604). For example, the received data may be encoded using the techniques discussed above with respect to Table 1, Table 2, and Fig. 2. The encoded data is transmitted across the channel (block 606). A receiving system at a receiving end of the channel receives the encoded data (block 608). The receiving system decodes the encoded data (block 610). For example, the received data may be encoded using the techniques discussed above with respect to Table 1, Table 2, and Figs. 3-5.

Fig. 7 is a flow diagram illustrating another procedure for encoding bytes of data to minimize current fluctuations from one byte of data to the next. Initially, data is received for transmission across a channel, bus, or other collection of conductors (block 702). Before the data is provided to the output drivers, the data is analyzed to determine the signal content required to transmit the data across the channel (block 704). Next, the procedure 700 determines whether the signal

current required to transmit the data falls within a predetermined range (block 706).

In a 4-level signaling system (such as 4-PAM), the possible range of current values for each symbol is $0i$, $1i$, $2i$, or $3i$. For eight symbols of a 4-PAM byte, the possible range of current required to transmit the eight symbols is $0i$ to $24i$ (i.e., 8 symbols \times $3i$). In a particular implementation, the transmission system designer has selected $8i$ to $16i$ as the preferred range for the total current required to transmit the data. This preferred range represents the middle third of the range $0i$ to $24i$. In alternate embodiments, the preferred range may be the first third of the range $0i$ to $24i$ (i.e., $0i$ to $8i$) or the last third of the range $0i$ to $24i$ (i.e., $16i$ to $24i$).

Referring again to Fig. 7, if the signal current required to transmit the data is within the predetermined range, then procedure 700 branches from block 706 to block 708, where the data is transmitted without encoding. If the signal current required to transmit the data is not within the predetermined range, then the procedure 700 continues from block 706 to block 710, where the data is encoded such that the signal current required to transmit the data is within the predetermined range. Next, the encoded data is transmitted along with an additional signal (2 bits) that indicates how the data was encoded (block 712). This additional signal is required to indicate to the decoder how to decode the encoded byte. Each successive data transmission may be encoded in a different manner. Therefore, an additional signal is transmitted along with each data transmission. In an alternate implementation, this additional signal is also transmitted if the data is not encoded. In this implementation, the additional signal provides an indication to the decoder that the data does not require any decoding.

1 When encoding a byte of data, each of the four two-bit symbols in the byte
2 is incremented by 1, 2, or 3 using modulo 4 arithmetic until the signal current
3 required to transmit the byte of data falls within the predetermined range. The
4 additional signal sent with the byte of data indicates whether the encoder
5 incremented the byte by 1, 2, or 3, or whether the encoder did not increment the
6 byte (i.e., the byte of data was not encoded).

7 The encoding process first determines whether incrementing each symbol
8 in the byte of data by one would result in a byte of data requiring a signal current
9 that is within the predetermined range. If so, each symbol in the byte of data is
10 incremented by one and an additional two-bit signal "01" is appended to the
11 transmitted data to indicate to the decoder that the each symbol in the original byte
12 of data was incremented by one. If incrementing each symbol by one does not
13 provide an acceptable signal current, then the encoding process determines
14 whether incrementing each symbol by two would result in a byte of data requiring
15 a signal current that is within the predetermined range. If so, each symbol in the
16 byte of data is incremented by two and an additional signal "10" is appended to
17 the transmitted data to indicate to the decoder that each symbol in the original byte
18 of data was incremented by two. If incrementing each symbol by two does not
19 provide an acceptable signal current, then the encoding process increments each
20 symbol in the byte of data by three and an additional signal "11" is appended to
21 the transmitted data to indicate to the decoder that each symbol in the original byte
22 of data was incremented by three. If the original byte of data was not encoded,
23 then an additional signal "00" is appended to the transmitted data to indicate to the
24 decoder that the original byte of data was not encoded.

1 In an example of the encoding process, a byte of data to be transmitted is
2 00001230 (represented as 01101100 in binary) and the predetermined range of
3 signal current required to transmit the byte of data is 8i to 16i. The signal current
4 required to transmit this byte of data is 6i, which is the numerical bitwise sum of
5 the byte ($1i + 2i + 3i$). Since 6i is not within the predetermined range of 8i to 16i,
6 the byte of data will be encoded. Initially, the encoding process determines
7 whether incrementing each symbol by one will result in a byte of data requiring a
8 signal current that is within the predetermined range. In this example,
9 incrementing each symbol by one produces 11112301 (0 increments to 1, 1
10 increments to 2, 2 increments to 3, and 3 increments to 0 with the carry being
11 discarded). The signal current required to transmit this encoded byte of data is 10i
12 ($1i + 1i + 1i + 1i + 2i + 3i + 1i$), which is within the predefined range of 8i to 16i.
13 Therefore, this encoded byte of data is transmitted along with an additional two-bit
14 signal ("01") indicating that each symbol of the original byte of data was
15 incremented by one. Thus, the resulting nine symbol transmission is 111123011.
16 The decoder then decodes the received encoded byte by decrementing each
17 symbol in the byte by one.

18 Fig. 8 illustrates a system 800 capable of implementing the encoding
19 procedure discussed with respect to Fig. 7. Each of the eight symbol lines (A0 –
20 A7) is coupled to three incrementers 802, 804, and 806. Incrementer 802
21 increments each symbol by one, incrementer 804 increments each symbol by two,
22 and incrementer 806 increments each symbol by three. Each symbol line and the
23 output of each incrementer 802, 804, and 806 are coupled to an input of a 4-to-1
24 multiplexer 808. The multiplexer 808 selects either the original symbol line, the
25 symbol line incremented by one, the symbol line incremented by two, or the

1 symbol line incremented by three, depending on the control signal received from a
2 4-to-2 encoder 818. The input selected by the multiplexer 808 is provided on the
3 output of the multiplexer. The eight outputs from the eight multiplexers 808 are
4 labeled Q0 – Q7.

5 Four window comparators 810, 812, 814, and 816 are coupled to the
6 encoder 818, and one of the symbol lines, or one of the three incrementers 802,
7 804, or 806. Each window comparator 810-816 has the same range of acceptable
8 values (i.e., 8 through 15), as established by the minimum and maximum inputs of
9 each window comparator. Window comparator 810 compares the non-
10 incremented value of the byte of data (A0-A7) to the defined range (8 – 15).
11 Window comparator 812 compares the eight symbols of data incremented by one
12 to the defined range. Window comparator 814 compares the eight symbols of data
13 incremented by two to the defined range. Window comparator 816 compares the
14 eight bits of data incremented by three to the defined range. Each window
15 comparator 810-816 sums the eight received signal lines before the window
16 comparison is performed. Only one of the window comparator outputs will be
17 asserted for any set of inputs.

18 Each window comparator 810-816 receives data on a bus 820, 822, 824,
19 and 826, respectively. Each bus 820-826 includes eight signal lines. The seven
20 diagonal lines extending from each bus 820-826 represent the signal lines received
21 from the circuits associated with the other seven input signals (A0-A6). Thus, bus
22 820 includes signals {A7, A6, A5, A4, A3, A2, A1, A0}, bus 822 includes signals
23 {A7+1, A6+1, A5+1, A4+1, A3+1, A2+1, A1+1, A0+1}, bus 824 includes signals
24 {A7+2, A6+2, A5+2, A4+2, A3+2, A2+2, A1+2, A0+2}, and bus 826 includes
25 signals {A7+3, A6+3, A5+3, A4+3, A3+3, A2+3, A1+3, A0+3}.

Fig. 9 illustrates a system 900 capable of decoding data that was encoded by the system shown in Fig. 8. System 900 receives nine two-bit signals, labeled Q0-Q8. Eight summation devices 902-916, sum together two different two-bit signals. For example, summation device 902 sums signals Q0 and Q8, summation device 904 sums signals Q1 and Q8, and so forth. Each summation device 902-916 generates a two-bit output, labeled OUT1-OUT8, respectively. Each two-bit output represents the sum of the two input signals provided to the corresponding summation device.

The logic to encode and decode the data does not require any storage or synchronous clocking. Therefore, the logic's impact on the latency of the transmission system is limited to the gate delays of the encoder and the decoder. Since the incrementing is performed on a bit-by-bit basis, a highly parallel encoding structure can be used.

Thus, there has been described a system that supports multiple discrete signal levels on a signal path to allow the transmission of various symbols. In one embodiment, the system supports eight discrete signal levels on the signal path such that each symbol represents three bits. In another embodiment, the system supports M discrete signal levels on the signal path such that each symbol represents $\log_2 M$ bits.

Although the description above uses language that is specific to structural features and/or methodological acts, it is to be understood that the invention defined in the appended claims is not limited to the specific features or acts described. Rather, the specific features and acts are disclosed as exemplary forms of implementing the invention.